Proving the Shalls:
Requirements, Proofs, and Model-Based Development

Dr. Steven P. Miller

Advanced Computing Systems
Rockwell Collins
400 Collins Road NE, MS 108-206
Cedar Rapids, Iowa 52498
spmiller@rockwellcollins.com

Rockwell Collins

Outline of Presentation

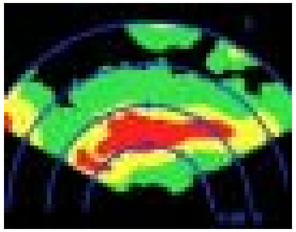
- **Introduction**
- **Overview of Our Approach**
- **Application to FGS Mode Logic**
- **Recent Applications**
- **Observations on Modeling**

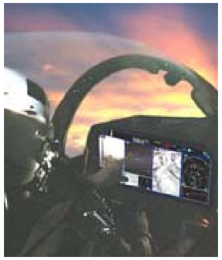
Who Are We?

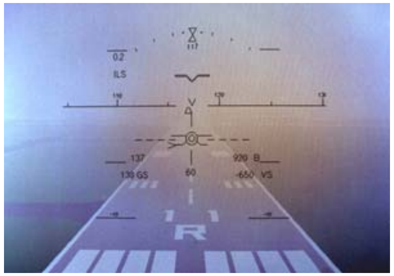
ADVANCED COMPUTING SYSTEMS

A World Leader In Aviation Electronics And Airborne/ Mobile Communications Systems For Commercial And Military Applications

- Communications
- Navigation
- Automated Flight Control
- Displays / Surveillance
- Aviation Services
- In-Flight Entertainment
- Integrated Aviation Electronics
- Information Management Systems



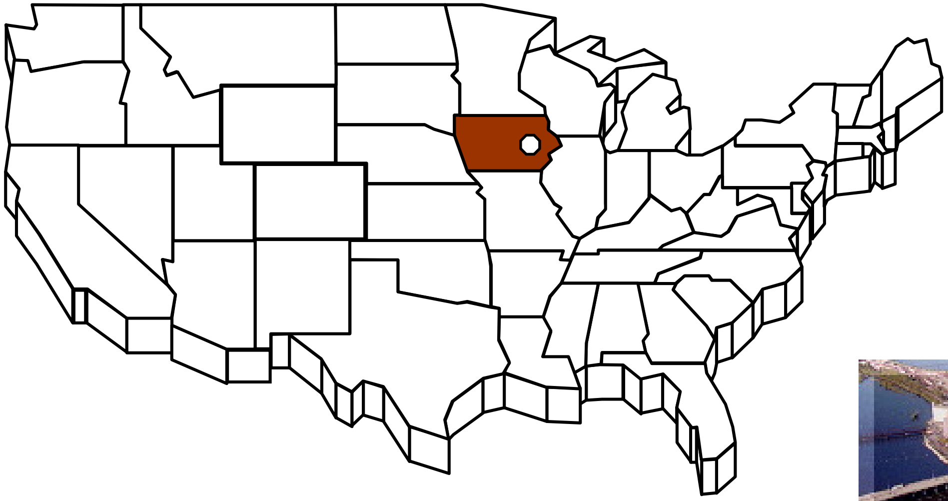




Rockwell Collins

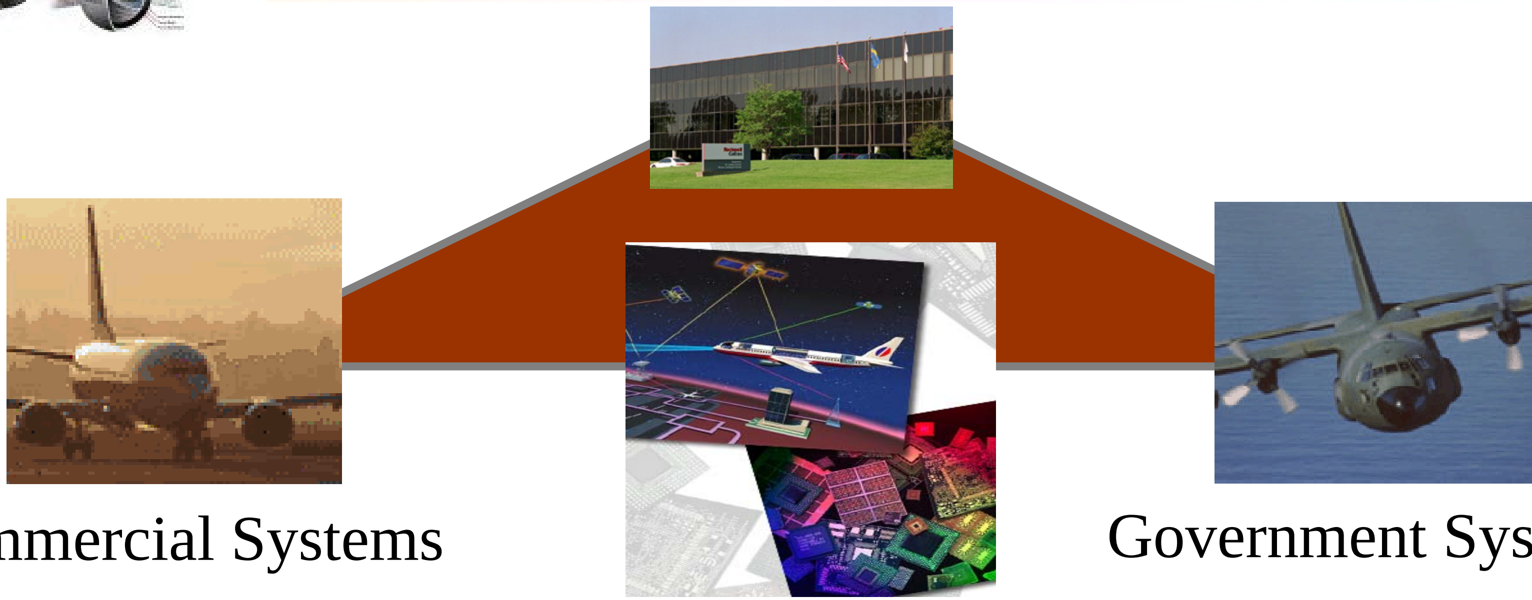
Headquartered in Cedar Rapids, Iowa

16,000 Employees Worldwide



RCI Advanced Technology Center

ADVANCED COMPUTING SYSTEMS



Commercial Systems

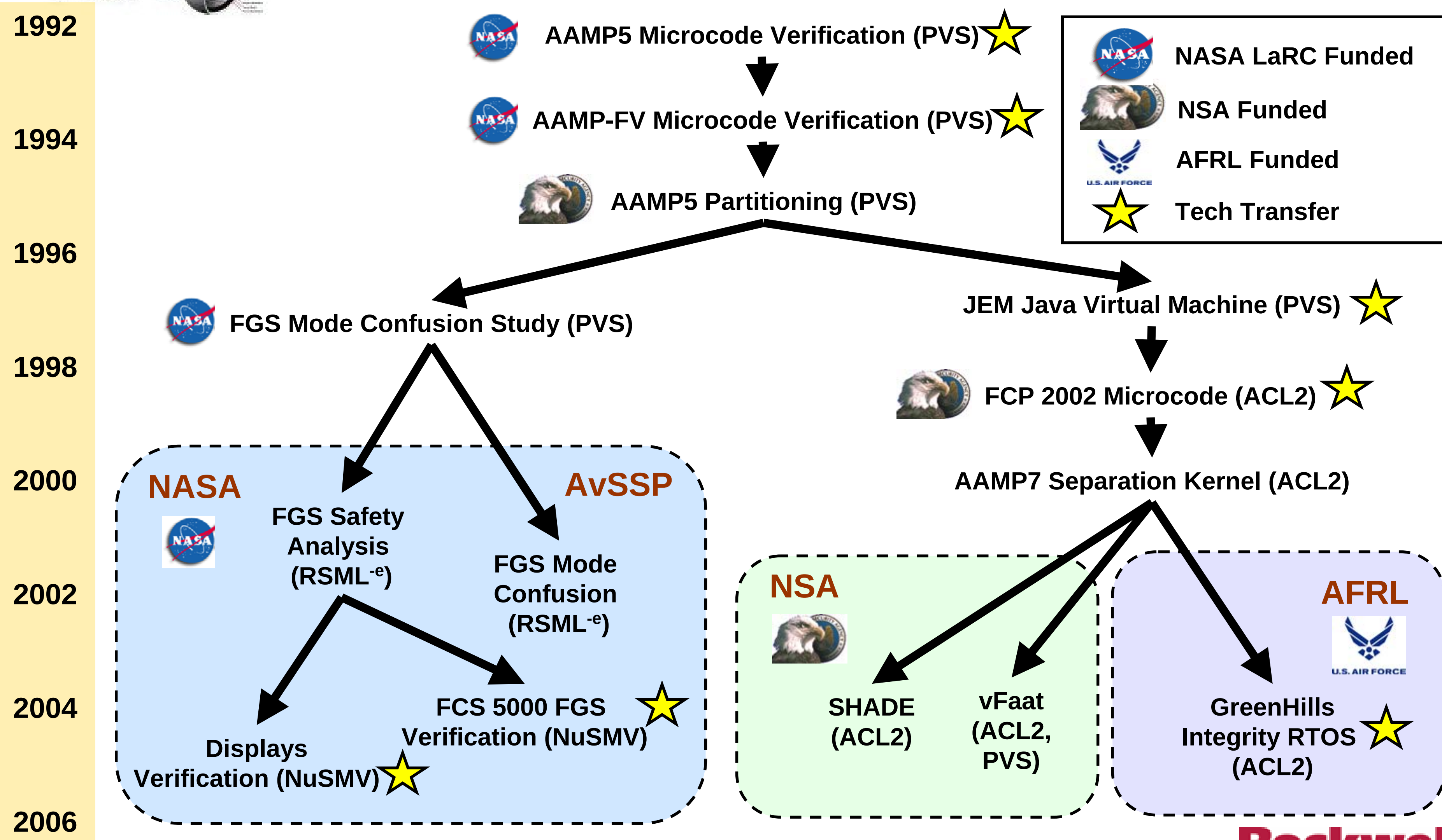
Government Systems

Advanced Technology Center

- The Advanced Technology Center (ATC) identifies, acquires, develops and transitions value-driven technologies to support the continued growth of Rockwell Collins.
- The Automated Analysis group applies mathematical tools and reasoning to the problem of producing high assurance systems.

Rockwell Collins

Automated Analysis Section



Methods and Tools for Flight Critical Systems Project

- **Five Year Project Started in 2001**
- **Part of NASA's Aviation Safety Program (Contract NCC-01001)**
- **Funded by the NASA Langley Research Center and Rockwell Collins**
- **<u>Practical</u> Application of Formal Methods To Modern Avionics Systems**

Outline of Presentation

Introduction

Overview of Our Approach

Application to FGS Mode Logic

Recent Applications

Observations on Modeling

Convergence of Two Trends

Model-Based Development

Automated Analysis

A Revolutionary Change in How We Design and Build Systems

Model-Based Development Examples

Company	Product	Tools	Specified & Autocoded	Benefits Claimed
Airbus	A340	SCADE With Code Generator	• 70% Fly-by-wire Controls • 70% Automatic Flight Controls • 50% Display Computer • 40% Warning & Maint Computer	• 20X Reduction in Errors • Reduced Time to Market
Eurocopter	EC-155/135 Autopilot	SCADE With Code Generator	• 90 % of Autopilot	• 50% Reduction in Cycle Time
GE & Lockheed Martin	FADEDC Engine Controls	ADI Beacon	• Not Stated	• Reduction in Errors • 50% Reduction in Cycle Time • Decreased Cost
Schneider Electric	Nuclear Power Plant Safety Control	SCADE With Code Generator	• 200,000 SLOC Auto Generated from 1,200 Design Views	• 8X Reduction in Errors while Complexity Increased 4x
US Spaceware	DCX Rocket	MATRIXx	• Not Stated	• 50-75% Reduction in Cost • Reduced Schedule & Risk
PSA	Electrical Management System	SCADE With Code Generator	• 50% SLOC Auto Generated	• 60% Reduction in Cycle Time • 5X Reduction in Errors
CSEE Transport	Subway Signaling System	SCADE With Code Generator	• 80,000 C SLOC Auto Generated	• Improved Productivity from 20 to 300 SLOC/day
Honeywell Commercial Aviation Systems	Primus Epic Flight Control System	MATLAB Simulink	• 60% Automatic Flight Controls	• 5X Increase in Productivity • No Coding Errors • Received FAA Certification

Does Model-Based Development Scale?

Airbus A380

Length	239 ft 6 in
Wingspan	261 ft 10 in
Maximum Takeoff Weight	1,235,000 lbs
Passengers	Up to 840
Range	9,383 miles

Systems Developed Using MBD

- **Flight Control**
- **Auto Pilot**
- **Fight Warning**
- **Cockpit Display**
- **Fuel Management**
- **Landing Gear**
- **Braking**
- **Steering**
- **Anti-Icing**
- **Electrical Load Management**

Outline of Presentation

Introduction

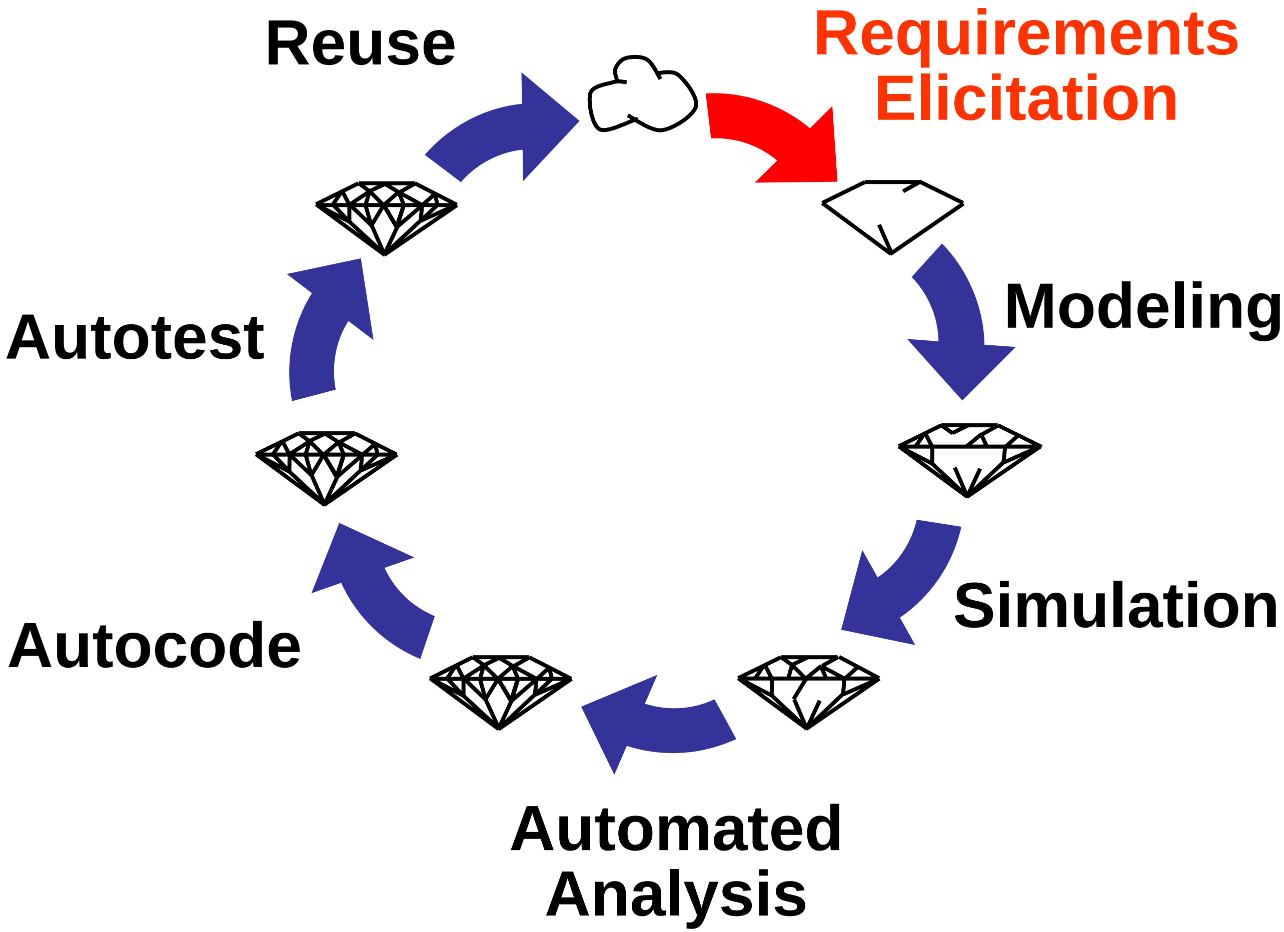
Overview of Our Approach

Application to FGS Mode Logic

Recent Applications

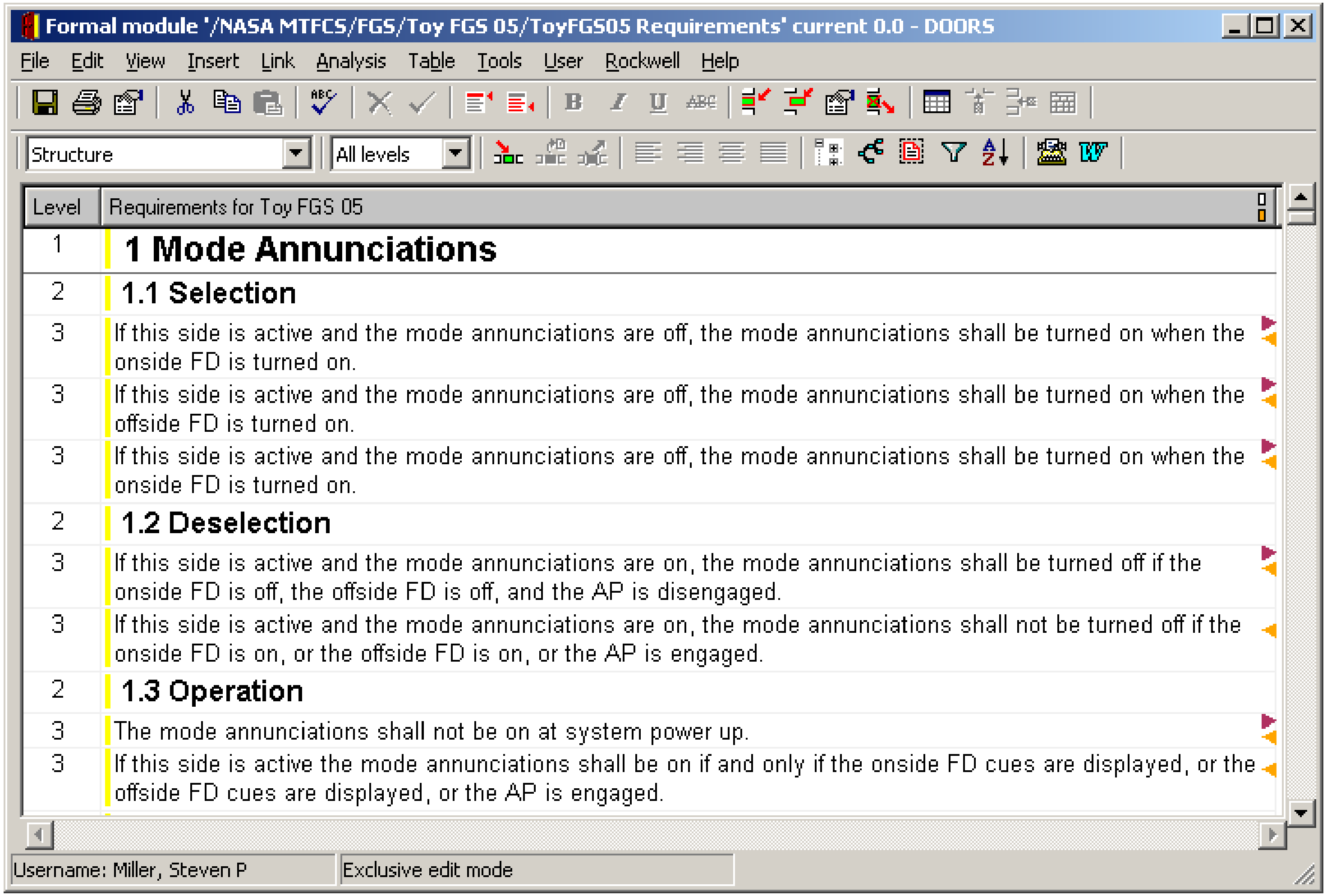
Observations on Modeling

Flight Guidance System Mode Logic

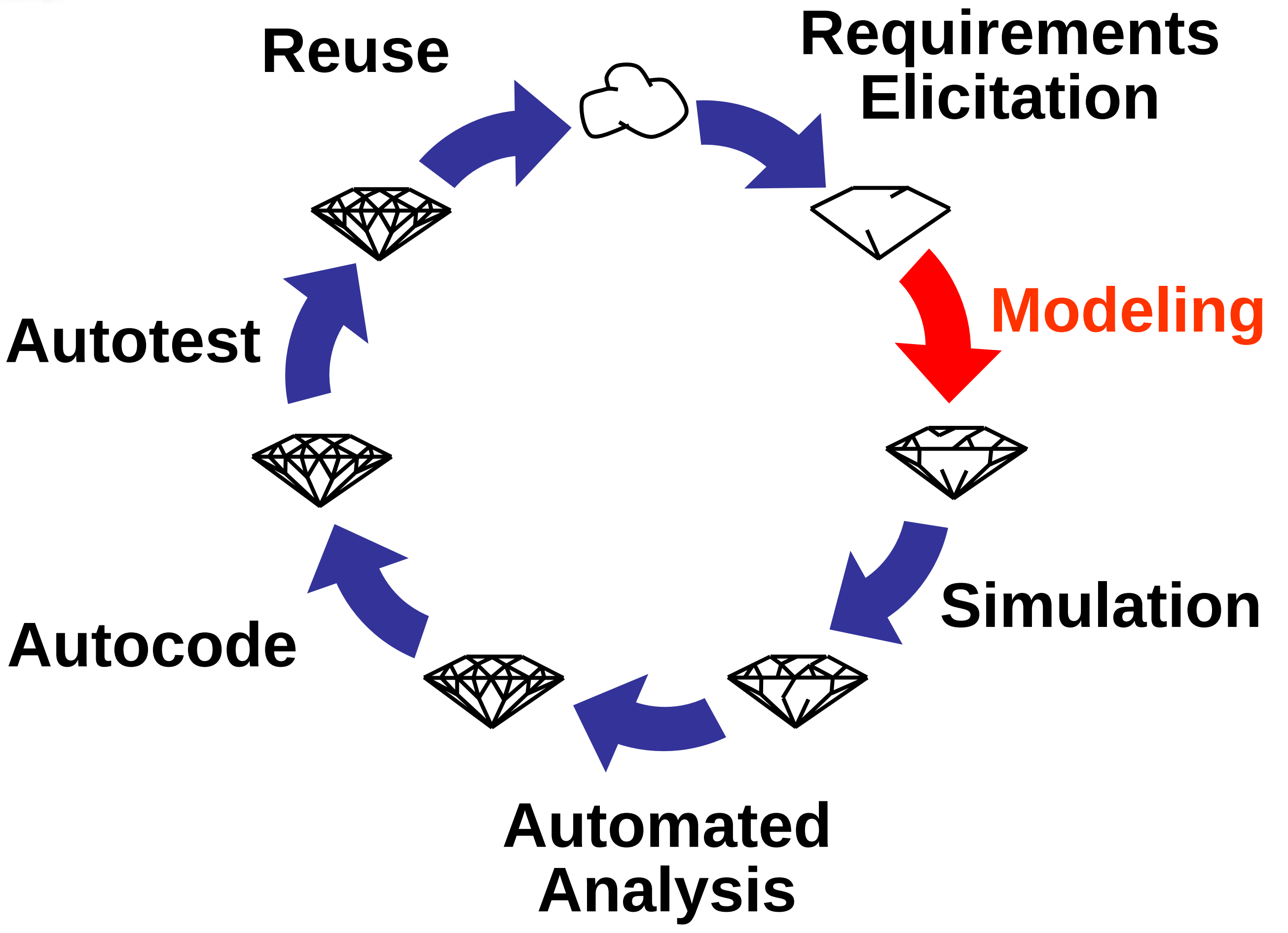


Captured Requirements as Shalls

ADVANCED COMPUTING SYSTEMS



Modeling


Reuse
Requirements Elicitation
Modeling
Simulation
Automated Analysis
Autocode
Autotest

Modeling Notations

Textual (Lustre, PVS, SAL, …)

```
node Thrust_Required(
   FG_Mode : FG_Mode_Type ;
   Airborne : bool ;
   In_Flare : bool ;
   Emergency_Descent : bool;
   Windshear_Warning : bool ;
   In_Eng_Accel_Zone : bool ;
   On_Ground : bool)
returns (IsTrue : bool) ;

let

IsTrue =
   (FG_Thrust_Mode(FG_Mode) and
    Airborne)
 or
   (Airborne and Emergency_Descent)
 or
   Windshear_Warning
 or
   ((FG_Mode = ThrottleRetard) and
     In_Flare)
 or
   (In_Eng_Accel_Zone and On_Ground) ;
tel ;
```

Tabular (RSML^{-e}, SCR)

2.3 Flight Director (FD)

The Flight Director (FD) displays the pitch and roll guidance commands to the pilot and copilot on the Primary Flight Display. This component defines when the Flight Director guidance cues are turned on and off.

Definitions of Values to be Imported

MACRO

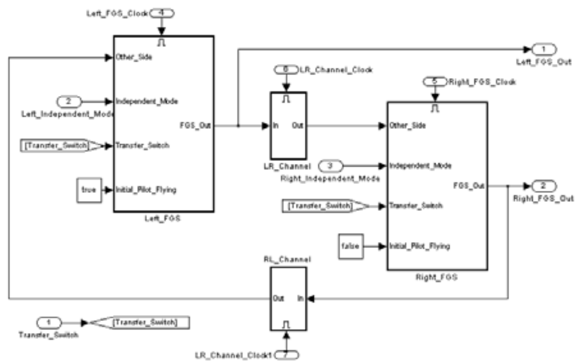
When_Turn_FD_On

Condition:

AND	OR						
When_FD_Switch_Pressed$_{m\text{-}96}$()	T	·	·	·	·	·	·
When(AP$_{v\text{-}129}$ =Engaged)	·	T	·	·	·	·	·
When(Overspeed$_{v\text{-}118}$)	·	·	T	·	·	·	·
When_GA_Switch_Pressed$_{m\text{-}102}$()	·	·	·	T	·	·	·
When_Lateral_Mode_Manually_Selected$_{m\text{-}23}$()	·	·	·	·	T	·	·
When_Vertical_Mode_Manually_Selected$_{m\text{-}24}$()	·	·	·	·	·	T	·
When_Pilot_Flying_Transfer$_{m\text{-}26}$()	·	·	·	·	·	·	T
Pilot_Flying$_{v\text{-}26}$ =THIS_SIDE$_{LEFT}$	·	·	·	·	·	·	T
Were_Modes_On$_{m\text{-}31}$()	·	·	·	·	·	·	T

Purpose: This event defines when the onside FD is to be turned on (i.e., displayed on the PFD).

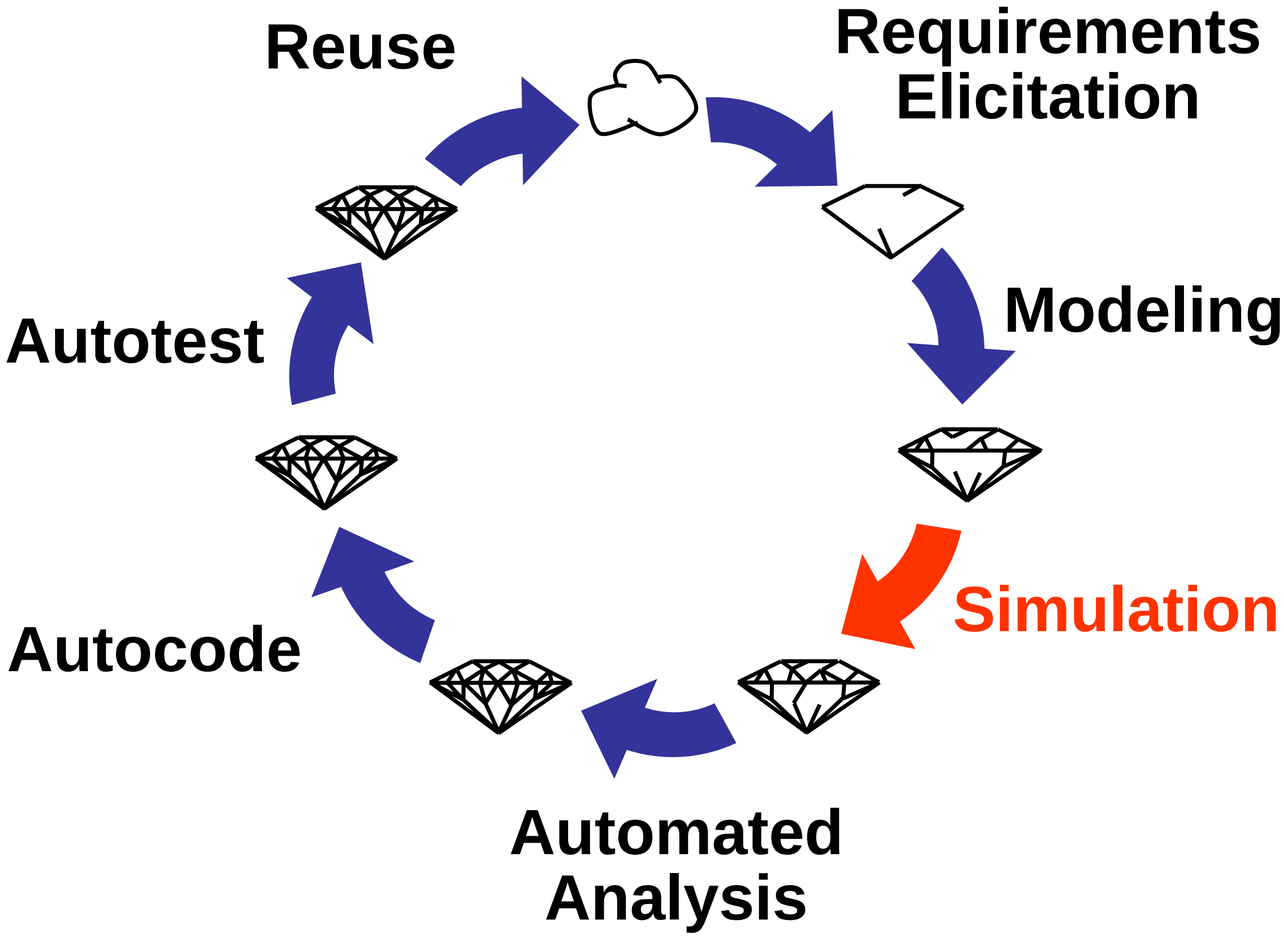
Graphical (Simulink, SCADE)



Rockwell Collins

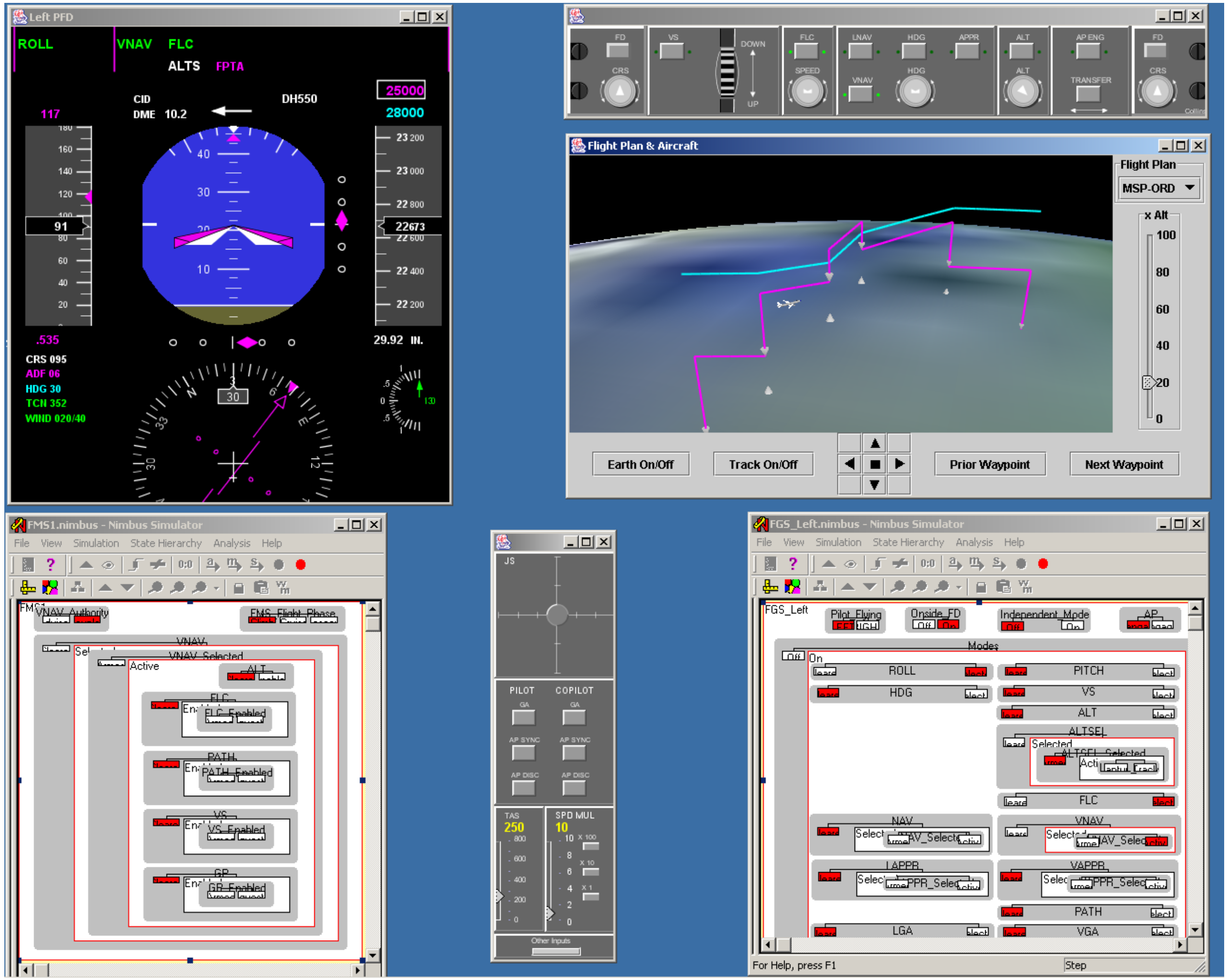
Simulation

ADVANCED COMPUTING SYSTEMS

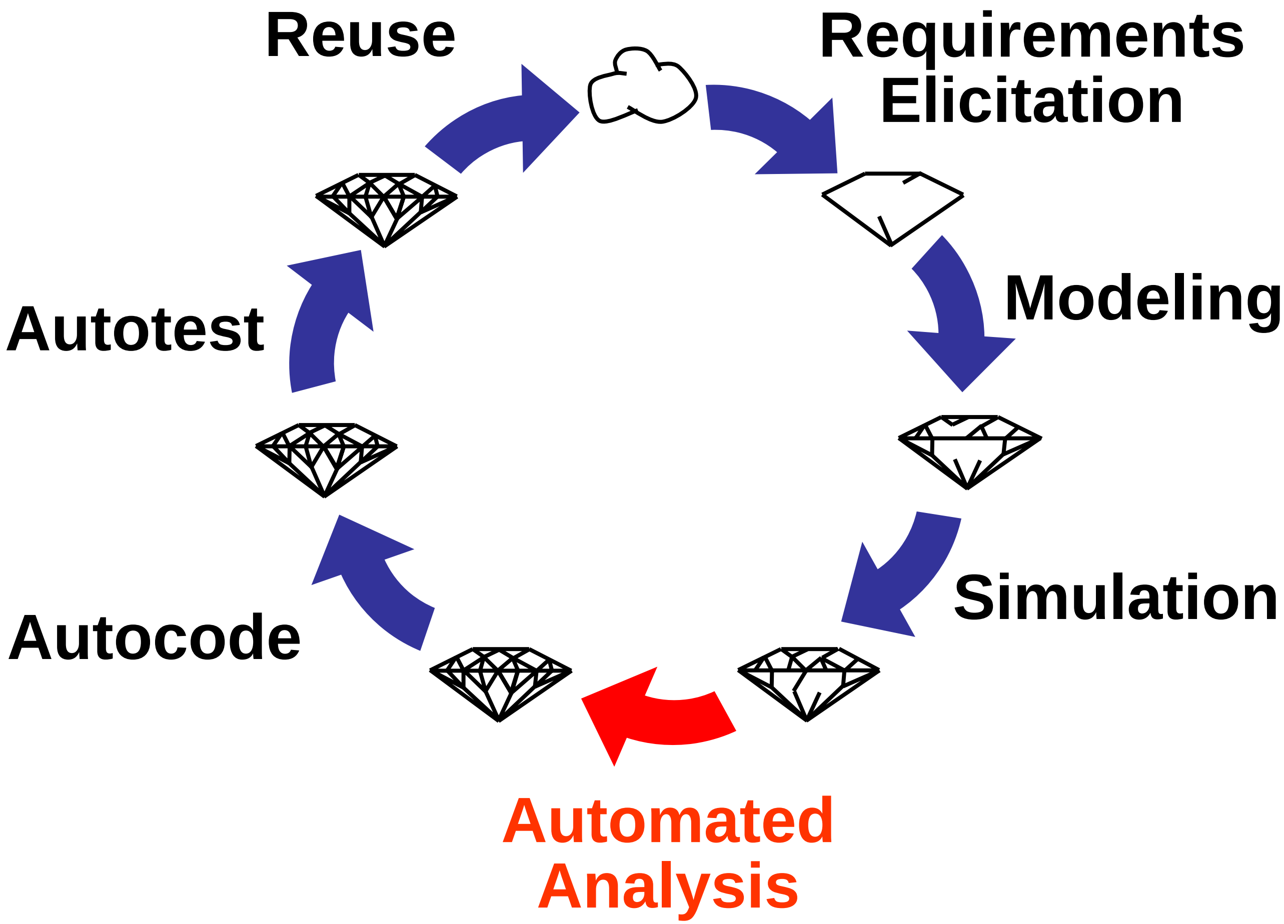


Simulation

ADVANCED COMPUTING SYSTEMS



Automated Analysis



Model Checkers

Theorem Provers

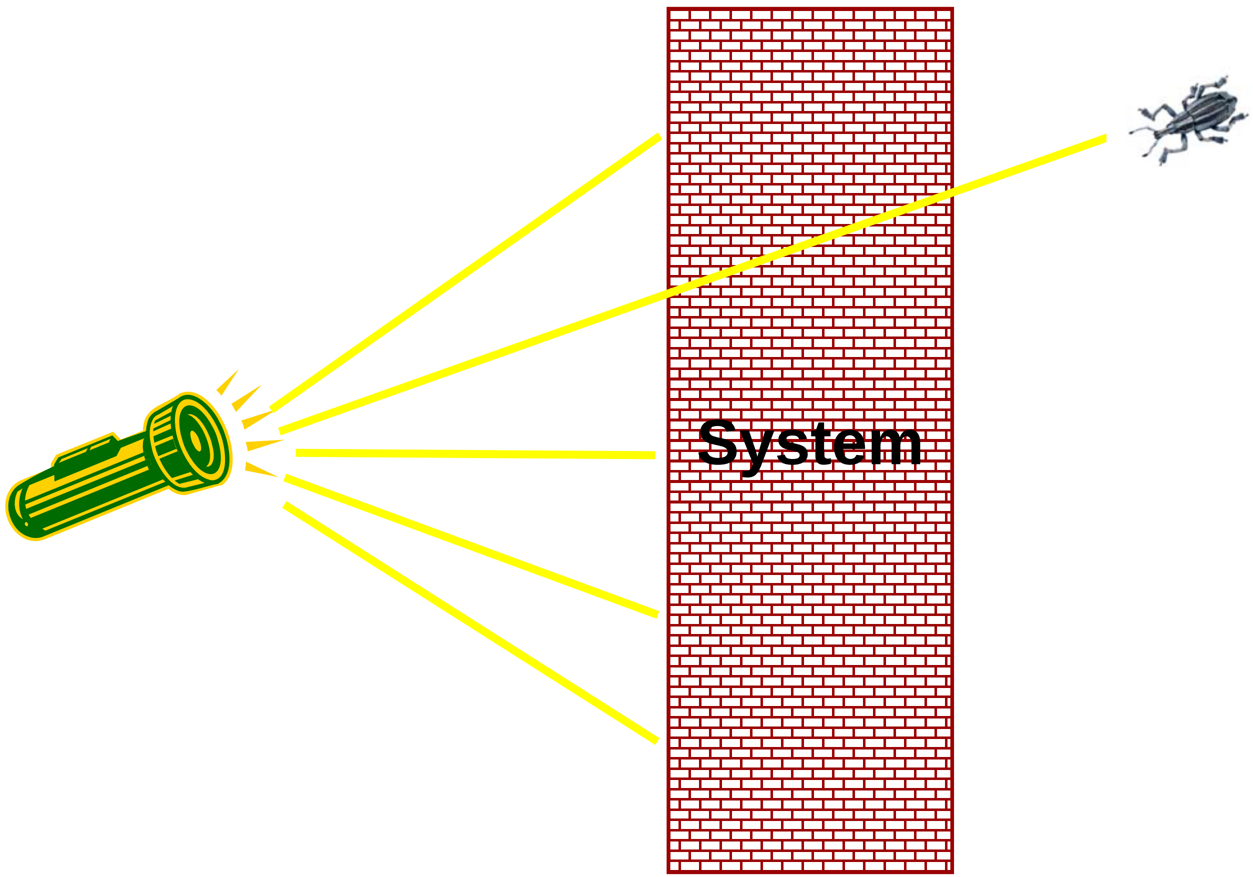
What Are Model Checkers?

- **Breakthrough Technology of the 1990's**
- **Widely Used in Hardware Verification (Intel, Motorola, IBM, …)**
- **Several Different Types of Model Checkers**
  - **Explicit, Symbolic, Bounded, Infinite Bounded, …**
- **Exhaustive Search of the Global State Space**
  - **Consider All Combinations of Inputs and States**
  - **Equivalent to Exhaustive Testing of the Model**
  - **Produces a Counter Example if a Property is Not True**
- **Easy to Use**
  - **"Push Button" Formal Methods**
  - **Very Little Human Effort Unless You're at the Tool's Limits**
- **Limitations**
  - **State Space Explosion (10^{100} – 10^{300} States)**

Advantage of Model Checking

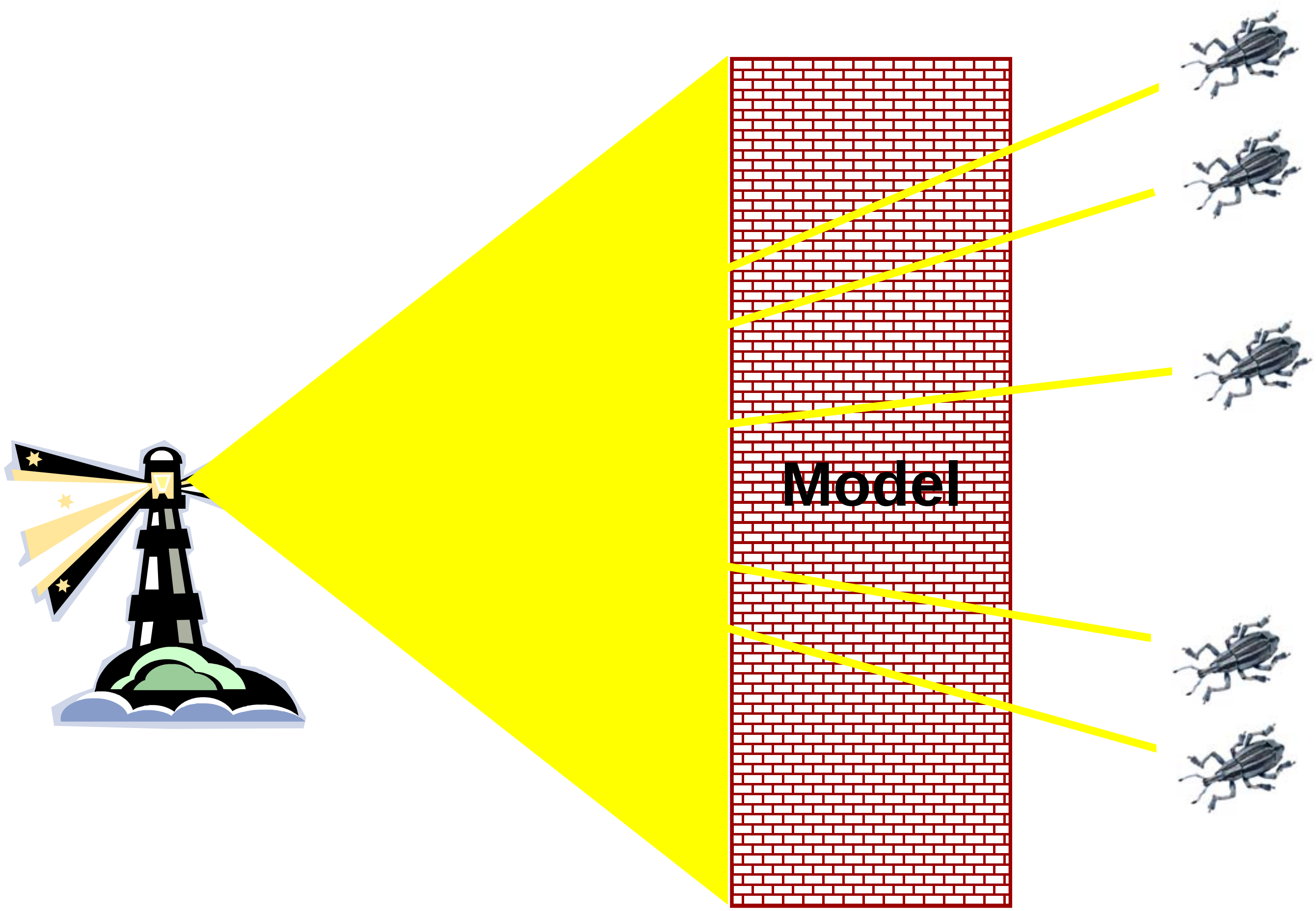
Testing Checks Only the Values We Select

Even Small Systems Have Trillions (of Trillions) of Possible Tests!



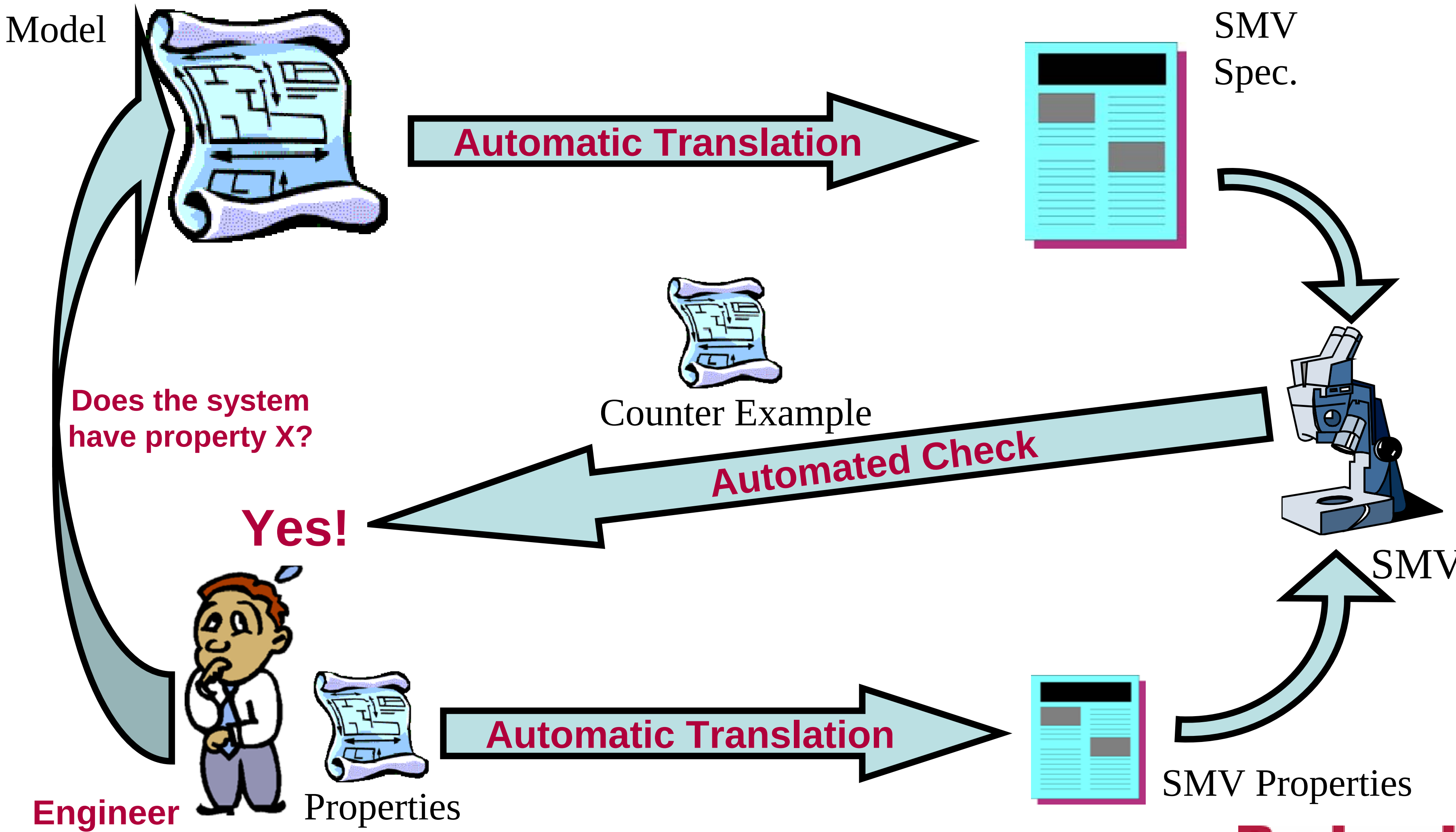
Advantage of Model Checking

Model Checker Tries Every Possible Input and State!



Model Checking Process

ADVANCED COMPUTING SYSTEMS



Rockwell Collins

Translated Shalls into SMV Properties

ADVANCED COMPUTING SYSTEMS

Formal module '/NASA MTFCS/FGS/Toy FGS 05/ToyFGS05 Requirements' current 0.0 - DOORS

File Edit View Insert Link Analysis Table Tools User Rockwell Help

SMV Plus | All levels

Ref. #	English Requirements	SMV Proof
1	**1 Mode Annunciations**	
1.1	**1.1 Selection**	
1.1.0-1	If this side is active and the mode annunciations are off, the mode annunciations shall be turned on when the onside FD is turned on.	SPEC AG((!Mode_Annunciations_On & !Onside_FD_On) -> AX((Is_This_Side_Active = 1 & Onside_FD_On) -> Mode_Annunciations_On))
1.1.0-2	If this side is active and the mode annunciations are off, the mode annunciations shall be turned on when the offside FD is turned on.	SPEC AG((!Mode_Annunciations_On & Offside_FD_On = FALSE) -> AX((Is_This_Side_Active = 1 & Offside_FD_On = TRUE) -> Mode_Annunciations_On))
1.1.0-3	If this side is active and the mode annunciations are off, the mode annunciations shall be turned on when the onside FD is turned on.	SPEC AG((!Mode_Annunciations_On & !Onside_FD_On) -> AX((Is_This_Side_Active = 1 & Onside_FD_On) -> Mode_Annunciations_On))
1.2	**1.2 Deselection**	
1.2.0-1	If this side is active and the mode annunciations are on, the mode annunciations shall be turned off if the onside FD is off, the offside FD is off, and the AP is disengaged.	SPEC AG(Mode_Annunciations_On -> AX((Is_This_Side_Active = 1 & !Onside_FD_On & Offside_FD_On = FALSE & !Is_AP_Engaged) -> !Mode_Annunciations_On))
1.2.0-2	If this side is active and the mode annunciations are on, the mode annunciations shall not be turned off if the onside FD is on, or the offside FD is on, or the AP is engaged.	SPEC AG(Mode_Annunciations_On -> AX((Is_This_Side_Active = 1 & (Onside_FD_On | Offside_FD_On = TRUE | Is_AP_Engaged)) -> Mode_Annunciations_On))
1.3	**1.3 Operation**	
1.3.0-1	The mode annunciations shall not be on at system power up.	SPEC (!Mode_Annunciations_On)
1.3.0-2	If this side is active the mode annunciations shall be on if and only if the onside FD cues are displayed, or the offside FD cues are displayed, or the AP is engaged.	SPEC AG(Is_This_Side_Active = 1 -> (Mode_Annunciations_On <-> (Onside_FD_On | Offside_FD_On = TRUE | Is_AP_Engaged)))

Username: Miller, Steven P | Exclusive edit mode

Rockwell Collins

Validate Requirements through Model Checking

ADVANCED COMPUTING SYSTEMS

```
xterm
-- specification AG (!Mode_Annunciations_On -> !Is_ALT_Selected) is true
-- specification AG (!Is_ALT_Selected -> AX (Is_ALT_Selected -> (This_Output_AltSel = 1 & This_Output_OPH_0))) is true
-- specification AG (Is_ALT_Selected <-> ALT_Lamp = ON) is true
-- specification AG (!Is_ALTSEL_Selected -> AX (((Modes = On & Is_This_Side_Active) & !Is_ALT_Selected) -> Is_ALTSEL_Selected)) is true
-- specification AG ((Is_ALTSEL_Selected & !Is_ALTSEL_Active) -> AX ((((Is_This_Side_Active & Modes = On) & m_When_ALTSEL_Capture_Cond_Met_Seen.result) & !m_Was_P
SA_Changed.result) -> Is_ALTSEL_Active)) is true
-- specification AG ((Is_ALTSEL_Active & !Is_ALTSEL_Track) -> AX ((((Is_This_Side_Active & Modes = On) & m_When_ALTSEL_Track_Cond_Met_Seen.result) & !m_Was_PSA_Ch
anged.result) -> Is_ALTSEL_Track)) is true
-- specification AG (Is_ALTSEL_Selected -> AX ((Is_This_Side_Active & Is_ALT_Selected) -> !Is_ALTSEL_Selected)) is true
-- specification AG (Is_ALTSEL_Active -> AX ((Is_This_Side_Active & m_When_Nonbasic_Vertical_Mode_Activated.result) -> !Is_ALTSEL_Active)) is true
-- specification AG (Is_ALTSEL_Active -> AX ((Is_This_Side_Active & m_When_Transfer_Switch_Pressed_Seen.result) -> !Is_ALTSEL_Active)) is true
-- specification AG (!Mode_Annunciations_On -> !Is_ALTSEL_Selected) is true
-- specification AG (!Is_ALTSEL_Track -> AX (Is_ALTSEL_Track -> (This_Output_AltselTrk = 1 & This_Output_OPH_0))) is true
-- specification AG (!Is_ALTSEL_Active -> AX (Is_ALTSEL_Active -> (This_Output_AltselAct = 1 & This_Output_OPH_0))) is true
-- specification AG (!Is_ALTSEL_Selected -> AX (Is_ALTSEL_Selected -> (This_Output_AltselSel = 1 & This_Output_OPH_0))) is true
-- specification AG (!Is_FLC_Selected -> AX (((Is_This_Side_Active & m_When_FLC_Switch_Pressed.result) & m_No_Higher_Event_Than_FLC_Switch_Pressed.result) -> Is_F
LC_Selected)) is true
-- specification AG (Is_FLC_Selected -> AX (((Is_This_Side_Active & m_When_FLC_Switch_Pressed.result) & m_No_Higher_Event_Than_FLC_Switch_Pressed.result) -> !Is_F
LC_Selected)) is true
-- specification AG (Is_FLC_Selected -> AX ((Is_This_Side_Active & m_When_Nonbasic_Vertical_Mode_Activated.result) -> !Is_FLC_Selected)) is true
-- specification AG (Is_FLC_Selected -> AX (((Is_This_Side_Active & m_When_Transfer_Switch_Pressed.result) & m_No_Higher_Event_Than_Transfer_Switch_Pressed.result
) -> !Is_FLC_Selected)) is true
-- specification AG (!Mode_Annunciations_On -> !Is_FLC_Selected) is true
-- specification AG (!Is_FLC_Selected -> AX (Is_FLC_Selected -> (This_Output_FlcSel = 1 & This_Output_OPH_0))) is true
-- specification AG (Is_FLC_Selected <-> FLC_Lamp = ON) is true
-- specification AG (!Is_This_Side_Active -> Mode_Annunciations_On = Offside_Modes_On = TRUE) is true
-- specification AG (!Is_This_Side_Active -> Is_ROLL_Selected = Offside_Roll_Selected = TRUE) is true
humboldt_Linux_spmiller>
```

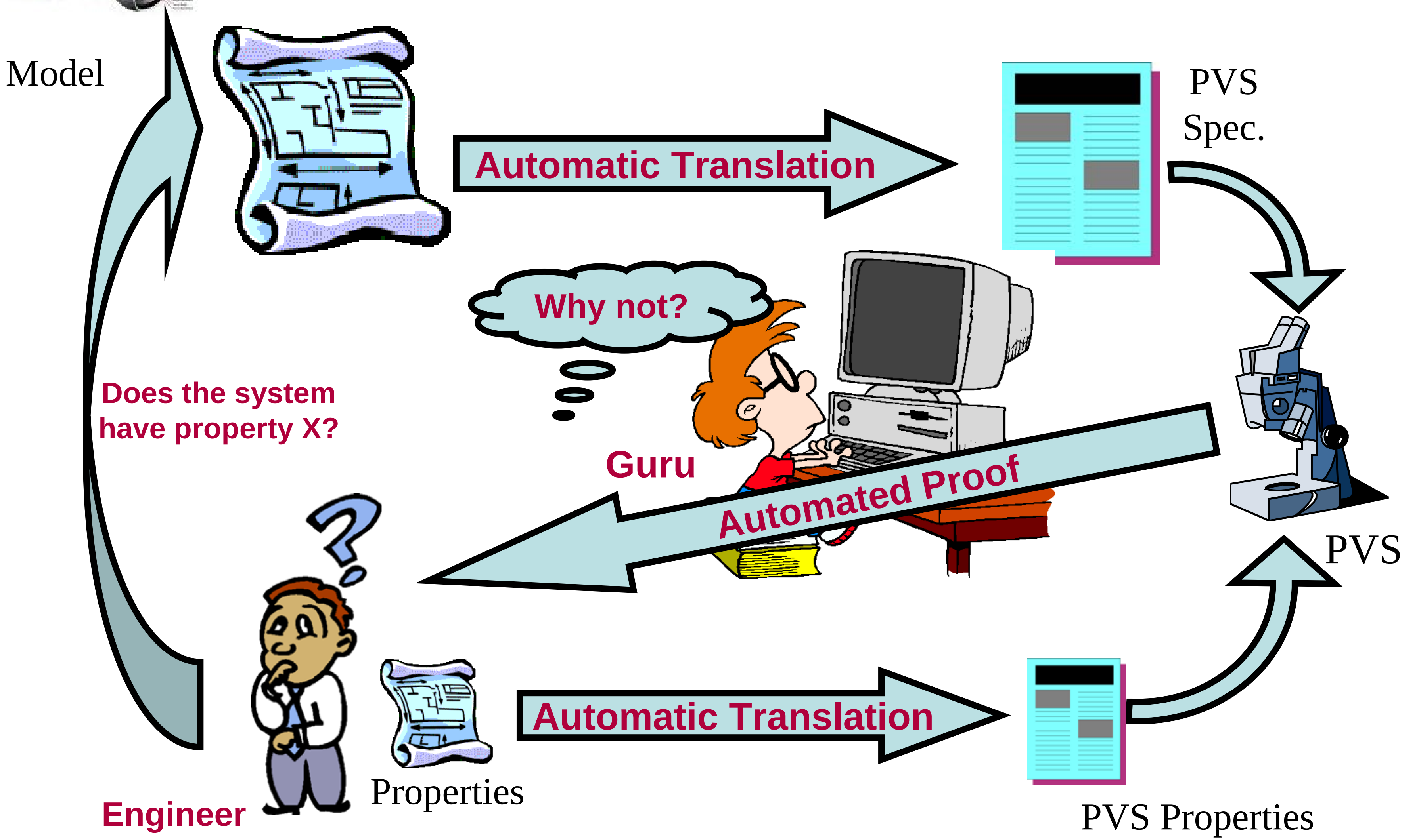
- Proved Over 280 Properties in Less Than an Hour
- Found Several Errors
- Some Were Errors in the Model
- Most Were Incorrect Shalls
- Revised the Shalls to Improve the Requirements

Rockwell Collins

What are Theorem Provers?

- **Available Since Late 1980's**
  - **Widely Used on Security and Safety-Critical Systems**
- **Use Rules of Inference to Prove New Properties**
  - **Also Consider All Combinations of Inputs and States**
  - **Also Equivalent to Testing with an Infinite Set of Test Cases**
  - **Generate An Unprovable Proof Obligation if a Property is False**
- **Not Limited by State Space**
  - **Applicable to Almost Any Formal Specification**
- **Limitations**
  - **Require Experience - About Six Months to Become Proficient**
  - **Constructing Proofs is Labor Intensive**

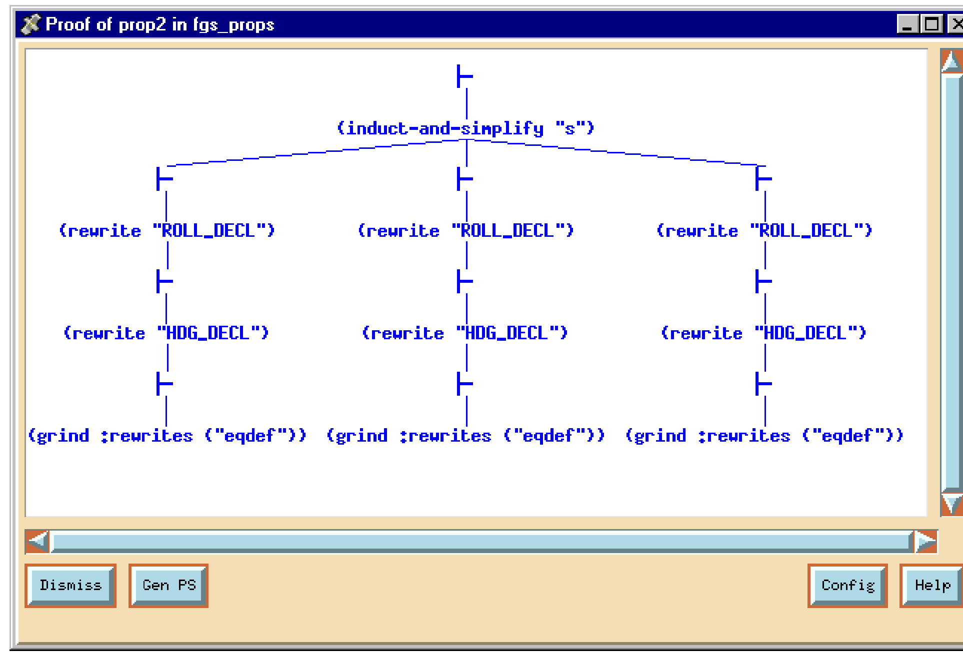
Theorem Proving Using PVS



Rockwell Collins

Validate Requirements Using Theorem Proving

- **Proved Several Hundred Properties Using PVS**
- **More Time Consuming that Model-Checking**
- **Use When Model-Checking Won't Work**



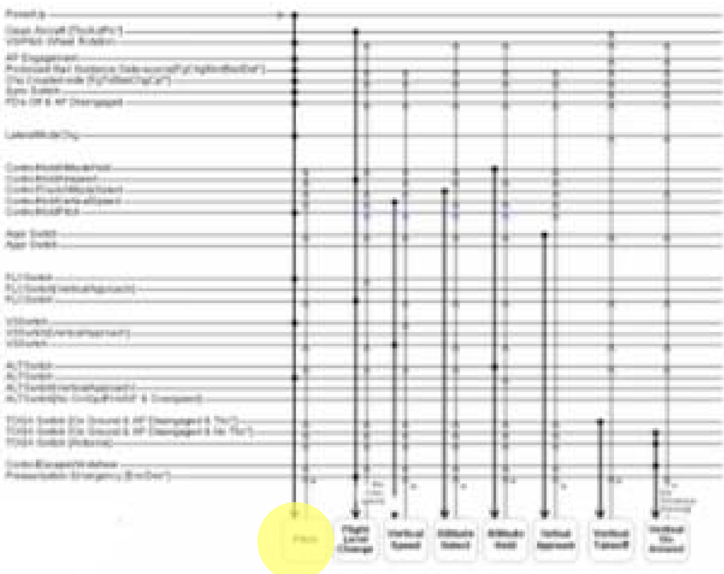
Rockwell Collins

Outline of Presentation

- Introduction
- Overview of Our Approach
- Application to FGS Mode Logic
- Recent Applications
- Observations on Modeling

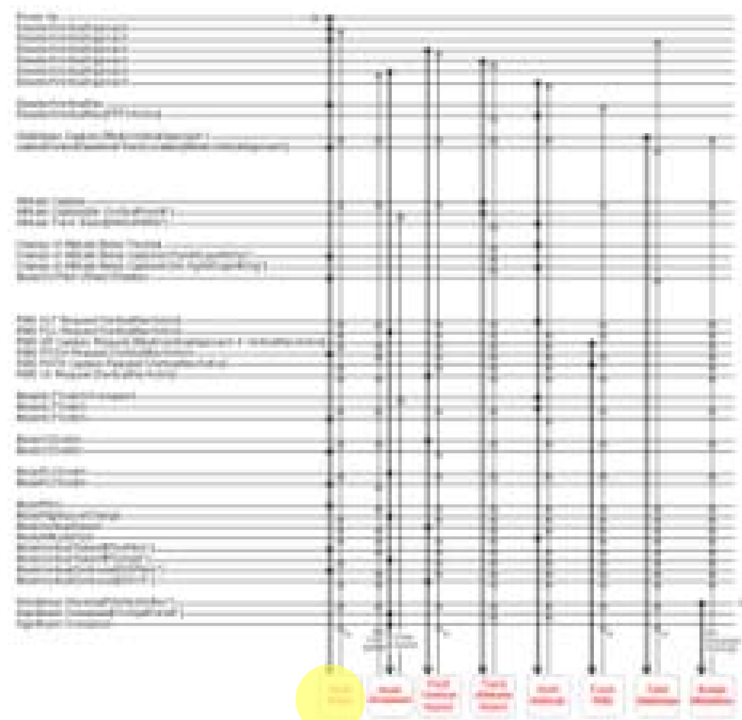
Example 1
FGS Mode Logic

Mode Controller A



6.8 x 10^{21} Reachable States

Mode Controller B



Requirement

Mode A1 => Mode B1

Counterexample Found in Less than Two Minutes!

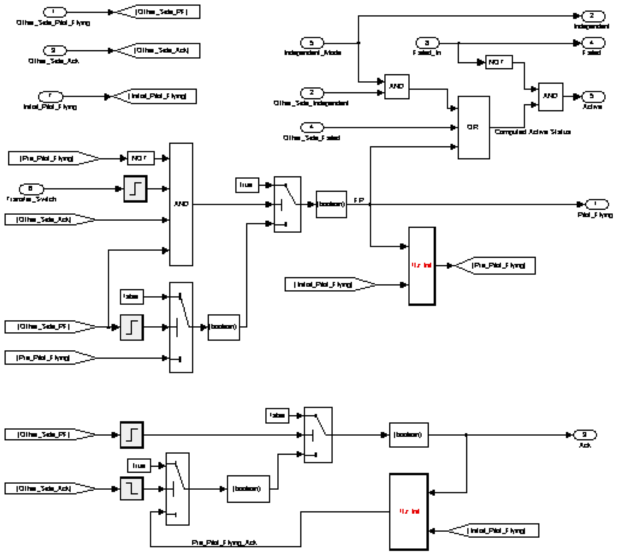
Found 26 Errors to Date

Example 2
Avionics Displays System

883 Subsystems

9,772 Simulink Blocks

2.9×10^{52} Reachable States



Requirement

Drive the Maximum Number of Display Units Given the Available Graphics Processors

Counterexample Found in 5 Seconds!

Checking Over 370 Properties
Found Over 60 Errors

Outline of Presentation

- Introduction
- Overview of Our Approach
- Application to FGS Mode Logic
- Recent Applications
- ➡ Observations on Modeling

Property (Constraint) Based Specifications

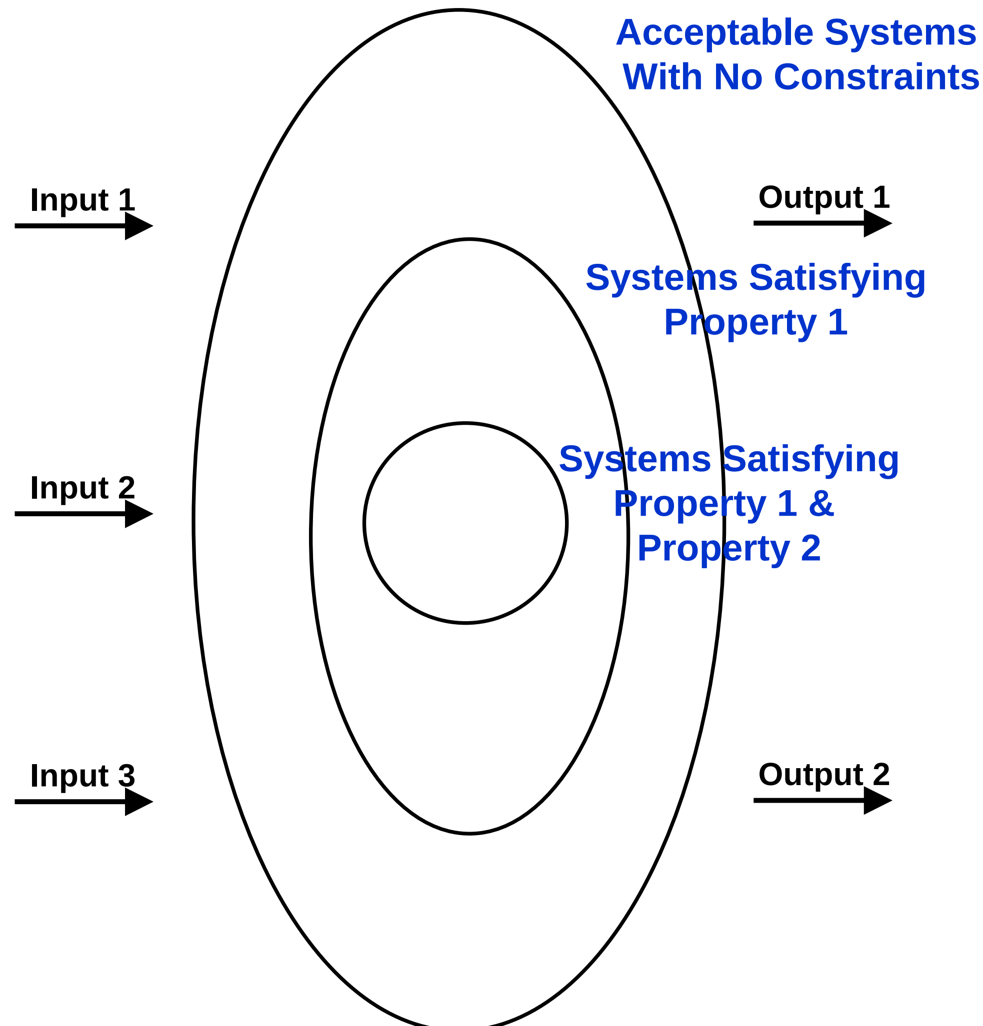
ADVANCED COMPUTING SYSTEMS

- **Define Acceptable Systems through Properties that**
  - Relate Outputs to Inputs
  - Constrain the Set of Acceptable Models
- **Make No Assumptions About Internal System Design**
- **Specify a Set of Acceptable Systems**

Property 1:
Output 2 > Input 1 + Input 2

Property 2:
Output 1 = Input 1/Input 3

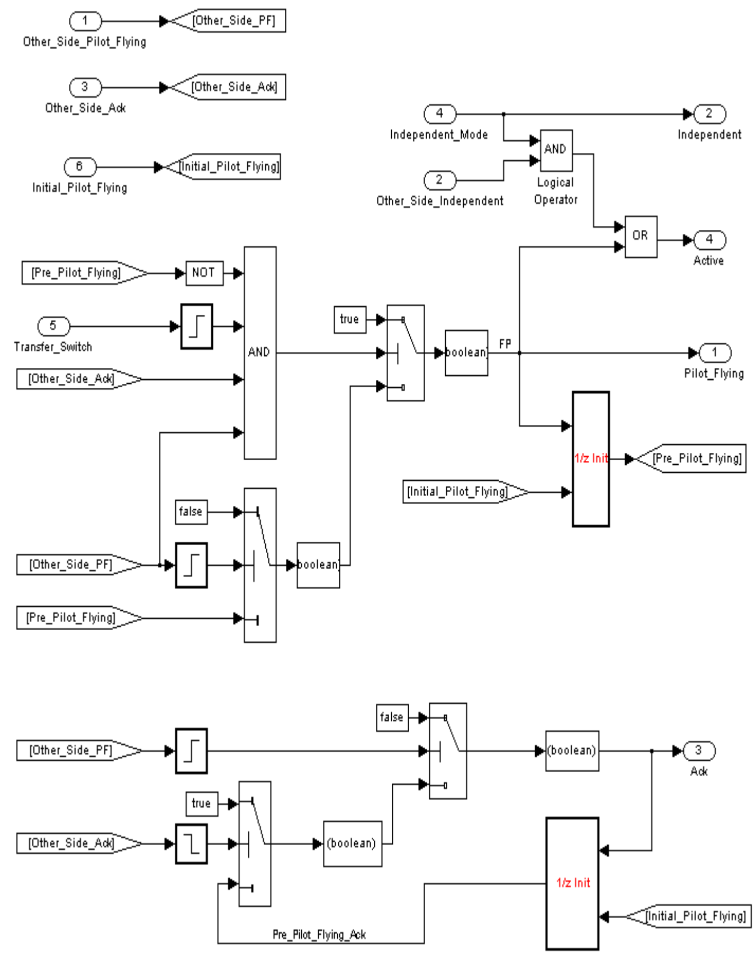
Property 3:
Output 2 = Input 1



Rockwell Collins

Constructive (Model) Based Specifications

- **Define Acceptable System(s) by <u>Constructing</u> a Model**
- **Start with a Set of Base Types**
  - **Booleans, Integers, Reals, …**
- **and a Set of Contructor Types**
  - **Records, Tuples, Arrays, ….**
- **Advantages**
  - **Intuitive**
  - **Models are Always Consistent**
  - **Models are Always Complete (a Behavior Defined for All Inputs)**
- **Disadvantages**
  - **Inherently Based on Internal Structure**
  - **Strongly Suggests a Design**
  - **Easy to Overconstrain Specification**



Rockwell Collins

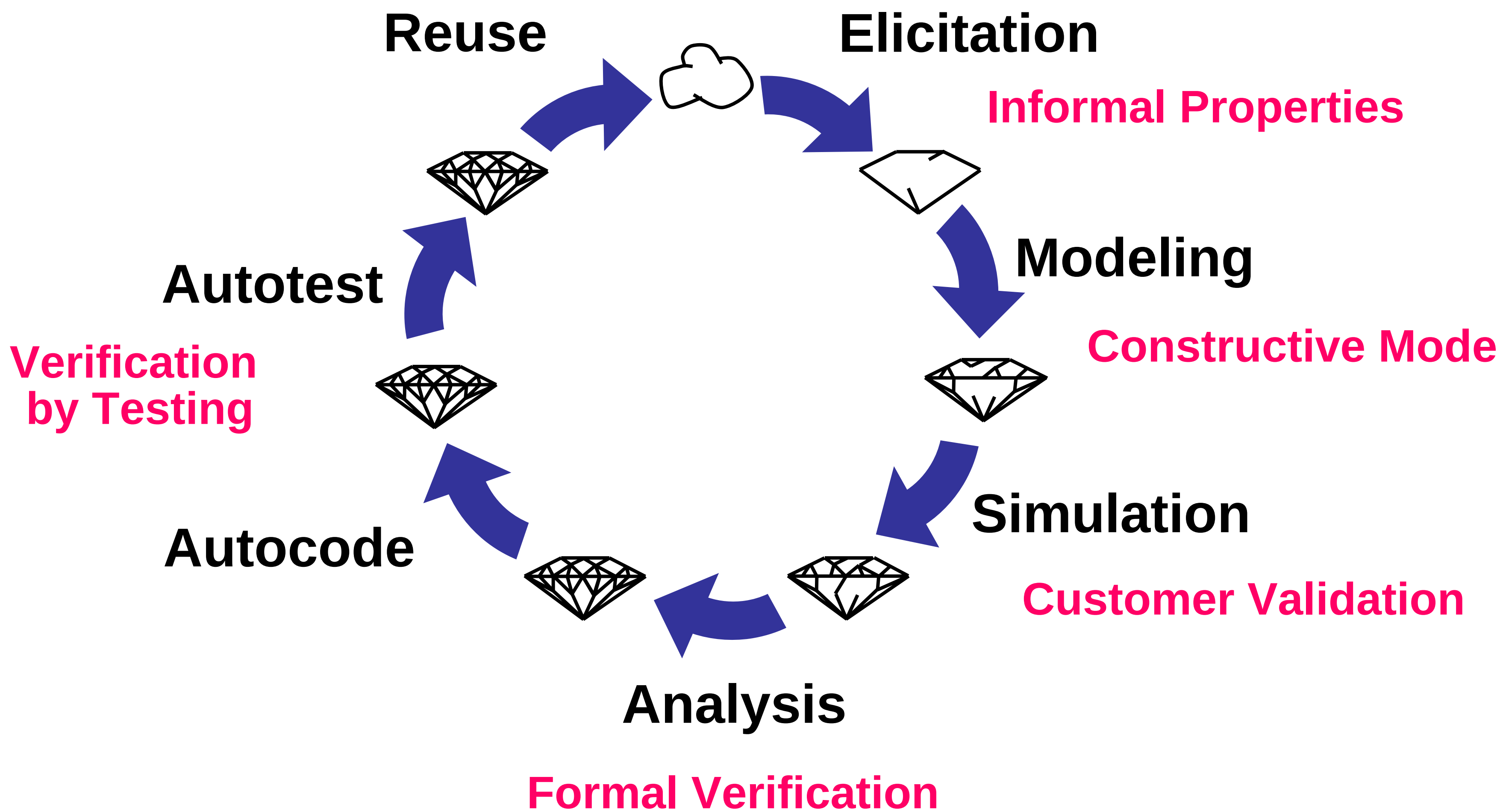
Strengths and Weaknesses of Specification Styles

	Natural Language	Property Based	Constructive Model
Ambiguity	Likely	Eliminated	Eliminated
Inconsistency	Likely	Possible	Eliminated
Incompleteness	Likely	Possible	Eliminated
Implementation Bias	Possible	Possible	Likely

Early ← Life Cycle → Late

Approach to Requirements Validation

ADVANCED COMPUTING SYSTEMS



Conclusions

- **Model-Based Development is the Industrial Use Formal Specification**
  - Providing the Modeling Language Has Well Defined Formal Semantics
- **Convergence of Model-Based Development and Formal Verification**
  - Key is to Get Engineers Producing Specifications that Can be Analyzed
- **Need Several Approaches to Formal Verification**
  - Model-Checking Because it is Simple and Easy to Use
  - Theorem Proving for When Model Checking isn’t Practical
- **Constructive Models are Useful**
  - Executable, Consistent, and Complete
  - Autogenerate Code and Test Cases
- **Shalls are Just Informal Property Based Specifications**
  - Easy Way to Elicit an Informal Description of the Requirements
  - Validate Constructive Model by Proving the Shalls!

For More Information

ADVANCED COMPUTING SYSTEMS

- Alan C. Tribble, Steven P. Miller, and David L. Lempia, *Software Safety Analysis of a Flight Guidance System*, NASA Contractor Report CR-2004-213004, March 2004, available at http://techreports.larc.nasa.gov/ltrs/dublincore/2004/cr/NASA-2004-cr213004.html.
- Alan C. Tribble and Steven P. Miller, Safety Analysis of Software Intensive Systems, IEEE Aerospace and Electronic Systems, Vol. 19, No. 10, pp. 21 - 26, October 2004.
- Steven P. Miller, Mats P.E. Heimdahl, and Alan C. Tribble, *Proving the Shalls*, in Proceedings of FM 2003: the 12th International FME Symposium, Pisa, Italy, Sept. 8-14, 2003.
- Alan C. Tribble, David D. Lempia, and Steven P. Miller, *Software Safety Analysis of a Flight Guidance System*, in Proceedings of the 21st Digital Avionics Systems Conference (DASC'02), Irvine, California, Oct. 27-31, 2002.

Rockwell Collins